Efficiency issues related to probability density function comparison

Patrick M. Kelly, Michael Cannon, Julio E. Barros

Computer Research and Applications Group, MS B-265
Los Alamos National Laboratory, Los Alamos, New Mexico

ABSTRACT

The **CANDID** project (Comparison Algorithm for Navigating Digital Image Databases) employs probability density functions (PDFs) of localized feature information to represent the content of an image for search and retrieval purposes. A similarity measure between PDFs is used to identify database images that are similar to a user-provided query image. Unfortunately, signature comparison involving PDFs is a very time-consuming operation. In this paper, we look into some efficiency considerations when working with PDFs. Since PDFs can take on many forms, we look into tradeoffs between accurate representation and efficiency of manipulation for several data sets. In particular, we typically represent each PDF as a Gaussian mixture (e.g. as a weighted sum of Gaussian kernels) in the feature space. We find that by constraining all Gaussian kernels to have principal axes that are aligned to the natural axes of the feature space, computations involving these PDFs are simplified. We can also constrain the Gaussian kernels to be hyperspherical rather than hyperellipsoidal, simplifying computations even further, and yielding an order of magnitude speedup in signature comparison. This paper illustrates the tradeoffs encountered when using these constraints.

Keywords: probability density functions, histograms, efficiency, distance measures, similarity measures

1 Introduction

Many content-based retrieval techniques for digital imagery use either a *feature vector* approach or a *feature histogram* approach to represent image content. That is, every object in an image (or every image by itself) is represented by a vector of specific feature measurements, or by a histogram of various feature value occurrences. When representing textures or shapes in an image, feature vector approaches are commonly used.[1,2] The texture of an image (or of a single object) is represented by a feature vector that can be compared to texture feature vectors from other database images using a weighted Euclidean distance, thereby allowing the retrieval of images with "similar" textures. In contrast to the feature vector approach, color content is typically described using a histogram. A histogram (in a three-dimensional color space) of the colors contained in each image (or in each distinct object) is computed, and an L_1, L_2, or an L_∞ distance is used to compare these color histograms.[3–5]

When using a feature-vector approach to describe image content, each component of a feature vector represents a single measurement taken over the entire image (or over an entire region of interest). Histogram approaches, on the other hand, allow us to represent the *distribution of localized features*, instead of restricting us to a single, global measurement. As an illustration of the added utility when using histograms, consider the problem of representing the color content of an image. A feature-vector approach might consist of computing an average red value, an average green value, and an average blue value for the entire image. The result would be a single feature vector of the "average" or "dominant" color characteristics in the image. A histogram approach, however, allows us to capture information about the overall distribution of colors in an image. We not only get a feel for the "average" or "dominant" color characteristics of the image, but we also retain information about the relative occurrences of the different color components, such as dark green versus light green. More information is represented when using the histogram approach, and it is therefore generally beneficial to favor histogram approaches over feature-vector based approaches, as long as we consider only the question of information representation and ignore questions about efficiency. Unfortunately, histogram approaches do not scale well with problem dimension. For color representation, where we are concerned with a three-dimensional RGB color space, we can easily divide the space into a discrete number of bins (e.g., 8x8x8 = 512 bins, or 16x16x16 = 4096 bins). As we consider higher-dimensional data, however, the number of bins required for accurate representation grows exponentially. Thus, histogram approaches do not produce viable solutions for problems concerning high-dimensional data. Another difficulty in using histograms is that they require a discretization of the feature space, which may not be easily obtainable.

The **CANDID** (**C**omparison **A**lgorithm for **N**avigating **D**igital **I**mage **D**atabases) project[6] employs parameterized representations of probability density functions (PDFs) to represent image content. Like histogram approaches, PDFs represent the *distribution of localized features*. But using parameterized representations of PDFs can circumvent some of the problems associated with histograms since "bins" are not explicitly designated in the feature space. Of course, computing probability density functions is much more expensive than computing histograms, so we are making a sacrifice in terms of computational cost. Similarly, comparing one PDF to another may be more expensive than comparing two histograms.

In this paper, we look at efficiency issues related to probability density function comparison. Specifically, we consider tradeoffs between accurate representation and efficiency of manipulation for **CANDID** signatures. We focus on the problem of speeding up image comparisons by reducing the computational complexity involved in distance calculations. We do not focus on the problem of efficient **CANDID** database indexing, which is discussed elsewhere in these proceedings.[7]

In the **CANDID** methodology, we typically represent each PDF as a Gaussian mixture (e.g. as a weighted sum of Gaussian kernels) in the feature space. If we constrain all Gaussian kernels to have principal axes that are aligned to the natural axes of the feature space, computations involving these PDFs are simplified. We can also constrain the Gaussian kernels to be hyperspherical rather than hyperellipsoidal, simplifying the computations even further. This paper illustrates the tradeoffs encountered when using these constraints (see Figure 1). Section 2 of this paper discusses the closed-form solution for computing distance and similarity measures with our **CANDID** signatures. Sections 3 and 4 look at constraints that can be used to simplify the calculations necessary to compare PDFs. Finally, Section 5 and Section 6 discuss some experimental results and conclusions.

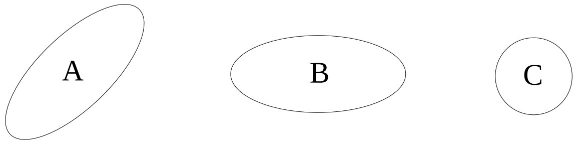

Figure 1: Constraining Gaussian Orientation And Shape. (A) Contour lines of a typical Gaussian kernel are hyperellipsoidal in shape. (B) Constraining the principal axes of the Gaussian to line up with the natural axes of the feature space reduces computational complexity for distance measures. (C) Constraining each kernel to be hyperspherical rather than hyperellipsoidal simplifies the computations even further.

2 Signature Representation and Comparison

CANDID employs probability density functions to represent image content in an approach that closely resembles the way histograms represent textual content in the N-gram approach to free-text document comparison.[8–10] The general idea is that we first compute several features (local color, texture, and/or shape) at every pixel in the image, and then compute a probability density function that describes the distribution of these features in an

N-dimensional feature space. This probability density function is our content signature for the given image. Of course, probability density function estimation is a large problem in itself; we attempt to estimate the probability density function as a Gaussian mixture. Each Gaussian distribution function is defined by a mean vector $\underline{\mu}_i$ (determining the position of the Gaussian) and a covariance matrix Σ_i (determining the shape and orientation of the Gaussian). A general data clustering routine can provide clusters for which for $\underline{\mu}_i$ and Σ_i can be obtained. We use the k-means clustering algorithm[11,12] followed by an optional cluster merging process.[13,14] A mean vector and covariance matrix are computed for each of the resultant clusters, and the associated Gaussian distribution function is weighted by the number of elements in the corresponding cluster. Any cluster having a singular covariance matrix is presently assumed to represent uninteresting data, and is therefore deleted from the data set and ignored in subsequent processing. Once a mixture of Gaussians has been identified, a signature over a specific N-dimensional feature space for image I can be represented as follows:

$$P_I(\underline{x}) \approx \sum_{i=1}^{K} w_i G_i(\underline{x}) \quad ; \quad G_i(\underline{x}) = (2\pi)^{-\frac{N}{2}} |\Sigma_i|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(\underline{x} - \underline{\mu}_i)^T \Sigma_i^{-1} (\underline{x} - \underline{\mu}_i) \right] \tag{1}$$

In the equations below, we differentiate between two different signatures, $P_{I_1}(\underline{x})$ and $P_{I_2}(\underline{x})$, with the following notation:

$$P_{I_1}(\underline{x}) = \sum_{i=1}^{K_1} w_i G_i(\underline{x}) \qquad P_{I_2}(\underline{x}) = \sum_{i=1}^{K_2} v_i F_i(\underline{x}) \tag{2}$$

We typically compare $P_{I_1}(\underline{x})$ and $P_{I_2}(\underline{x})$ using an L_2 distance measure $dist\,(I_1, I_2)$, or a normalized inner-product $sim\,(I_1, I_2)$, as defined below:

$$dist\,(I_1, I_2) = \left[\int_{\Re} (\, P_{I_1}(\underline{x}) - P_{I_2}(\underline{x})\,)^2 \, d\underline{x} \right]^{\frac{1}{2}} \tag{3}$$

$$sim\,(I_1, I_2) = \frac{\int_{\Re} P_{I_1}(\underline{x}) P_{I_2}(\underline{x}) d\underline{x}}{\left[\int_{\Re} P_{I_1}^2(\underline{x}) d\underline{x} \int_{\Re} P_{I_2}^2(\underline{x}) d\underline{x} \right]^{\frac{1}{2}}} \tag{4}$$

Using the signature representation given in Equation (1), these measures expand as follows:

$$\begin{aligned} dist\,(I_1, I_2) = & \left[\sum_{i=1}^{K_1} w_i^2 \int_{\Re} G_i^2(\underline{x})\, d\underline{x} + \sum_{i=1}^{K_2} v_i^2 \int_{\Re} F_i^2(\underline{x})\, d\underline{x} + 2 \sum_{i=1}^{K_1} \sum_{j=i+1}^{K_1} w_i w_j \int_{\Re} G_i(\underline{x}) G_j(\underline{x})\, d\underline{x} + \right. \\ & \left. 2 \sum_{i=1}^{K_2} \sum_{j=i+1}^{K_2} v_i v_j \int_{\Re} F_i(\underline{x}) F_j(\underline{x})\, d\underline{x} - 2 \sum_{i=1}^{K_1} \sum_{j=1}^{K_2} w_i v_j \int_{\Re} G_i(\underline{x}) F_j(\underline{x})\, d\underline{x} \right]^{\frac{1}{2}} \end{aligned} \tag{5}$$

$$\begin{aligned} sim\,(I_1, I_2) = & \left(\sum_{i=1}^{K_1} \sum_{j=1}^{K_2} w_i v_j \int_{\Re} G_i(\underline{x}) F_j(\underline{x}) d\underline{x} \right) \cdot \\ & \left(\sum_{i=1}^{K_1} w_i^2 \int_{\Re} G_i^2(\underline{x}) d\underline{x} + 2 \sum_{i=1}^{K_1} \sum_{j=i+1}^{K_1} w_i w_j \int_{\Re} G_i(\underline{x}) G_j(\underline{x}) d\underline{x} \right)^{-\frac{1}{2}} \cdot \\ & \left(\sum_{i=1}^{K_2} v_i^2 \int_{\Re} F_i^2(\underline{x}) d\underline{x} + 2 \sum_{i=1}^{K_2} \sum_{j=i+1}^{K_2} v_i v_j \int_{\Re} F_i(\underline{x}) F_j(\underline{x}) d\underline{x} \right)^{-\frac{1}{2}} \end{aligned} \tag{6}$$

These measures both contain $O(K_1^2 + K_2^2)$ terms consisting of an infinite integral over the product of two Gaussians. These integrals can be computed as follows:

$$\int_{\Re} G_i(\underline{x})\; G_j(\underline{x})\; d\underline{x} \quad = \quad (2\pi)^{-\frac{N}{2}}\, |\Sigma_i + \Sigma_j|^{-\frac{1}{2}} \exp\left[-\frac{1}{2}(c_1 + c_2)\right] \tag{7}$$

where c_1 and c_2 are given by:

$$c_1 \quad = \quad \underline{\mu}_i{}^T \Sigma_i^{-1} \underline{\mu}_i \; + \; \underline{\mu}_j{}^T \Sigma_j^{-1} \underline{\mu}_j \tag{8}$$

$$c_2 \quad = \quad -(\underline{\mu}_i{}^T \Sigma_i^{-1} + \underline{\mu}_j{}^T \Sigma_j^{-1}) \cdot (\Sigma_i^{-1} + \Sigma_j^{-1})^{-1} \cdot (\underline{\mu}_i{}^T \Sigma_i^{-1} + \underline{\mu}_j{}^T \Sigma_j^{-1})^T \tag{9}$$

3 Constrained Gaussian Orientation

In an effort to simplify the calculations and speed up **CANDID** signature comparison, we will constrain all covariance matrices in the previous section to be diagonal. That is, every off-diagonal element will be 0. Note that all diagonal elements of these covariance matrices are denoted σ_{ik}^2 which corresponds with the fact that each value represents a *variance* in dimension k for the associated Gaussian.

$$\Sigma_i = \begin{bmatrix} \sigma_{i1}^2 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_{i2}^2 & 0 & & 0 \\ 0 & 0 & \sigma_{i3}^2 & & 0 \\ \vdots & & & \ddots & 0 \\ 0 & 0 & 0 & 0 & \sigma_{iN}^2 \end{bmatrix} \qquad \Sigma_j = \begin{bmatrix} \sigma_{j1}^2 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_{j2}^2 & 0 & & 0 \\ 0 & 0 & \sigma_{j3}^2 & & 0 \\ \vdots & & & \ddots & 0 \\ 0 & 0 & 0 & 0 & \sigma_{jN}^2 \end{bmatrix} \tag{10}$$

Since Gaussian kernels with diagonal covariance matrices have their principal axes aligned with the natural axes of the feature space (as shown in picture (B) of Figure 1), we will refer to this constraint as the *axis alignment constraint.* When we constrain the form of our covariance matrices in this manner, the evalutation of Equation (7) is simplified substantially:

$$|\Sigma_i + \Sigma_j|^{-\frac{1}{2}} \quad = \quad \left[\prod_{k=1}^{N} (\sigma_{ik}^2 + \sigma_{jk}^2)\right]^{-\frac{1}{2}} \tag{11}$$

$$c_1 \quad = \quad \sum_{k=1}^{N} \left(\frac{\mu_{ik}^2}{\sigma_{ik}^2} \; + \; \frac{\mu_{jk}^2}{\sigma_{jk}^2}\right) \tag{12}$$

$$c_2 \quad = \quad -\sum_{k=1}^{N} \left(\frac{\sigma_{ik}^2 \sigma_{jk}^2}{\sigma_{ik}^2 + \sigma_{jk}^2}\right) \left(\frac{\mu_{ik}}{\sigma_{ik}^2} + \frac{\mu_{jk}}{\sigma_{jk}^2}\right)^2 \tag{13}$$

These values are substantially less expensive to compute as compared to their counterparts for the non-constrained Gaussian case.

4 Hyperspherical Constraints

In an effort to simplify our calculations even further, we now constrain all covariance matrices to be represented by a single scalar value multiplied by the identity matrix. Again, these scalar values represent variances, and are denoted as squared values.

$$\Sigma_i = \sigma_i^2 I \qquad \Sigma_j = \sigma_j^2 I \tag{14}$$

Gaussian kernels of this type will have hyperspherical contour lines, as indicated in picture (C) of Figure 1. We therefore refer to this as a *hyperspherical constraint*. Evaluating Equation (7) is now simplified even further than it was in Section 3:

$$|\Sigma_i + \Sigma_j|^{-\frac{1}{2}} = \left(\sigma_i^2 + \sigma_j^2\right)^{-\frac{N}{2}} \tag{15}$$

$$c_1 = \frac{1}{\sigma_i^2}\sum_{k=1}^{N}\mu_{ik}^2 + \frac{1}{\sigma_j^2}\sum_{k=1}^{N}\mu_{jk}^2 \tag{16}$$

$$c_2 = -\left(\frac{1}{\sigma_i^2 + \sigma_j^2}\right)\left[\frac{\sigma_j^2}{\sigma_i^2}\sum_{k=1}^{N}\mu_{ik}^2 + \frac{\sigma_i^2}{\sigma_j^2}\sum_{k=1}^{N}\mu_{jk}^2 + 2\sum_{k=1}^{N}\mu_{ik}\mu_{jk}\right] \tag{17}$$

5 Experimental Results

Remotely-sensed data can be used to locate underground oil reserves, monitor pollution from large factories, and track the disappearance of the Earth's rain forests. A database containing imagery collected by airborne sensors will prove much more valuable if scientists can access the data by searching on different attributes of image content instead of only being able to retrieve data by searching on associated textual metadata. The ability to automatically locate areas having similar ground cover will enable scientists to search through terabyte-sized image databases in order to study environmental problems. As an example, if a coniferous forest in Oregon is rapidly disappearing for no apparent reason, then other areas around the world having similar vegetation can be retrieved to see if they are experiencing the same problem. Scientists would then know if this was a global phenomenon or if local conditions were to blame.

We have applied **CANDID** to the problem of retrieving multispectral satellite data (Landsat TM data) from a database. This enables queries such as, "Show me all images of areas with landcover similar to this example." As an experiment, we created a database containing 120, 512 × 512, 6-banded images (the thermal infrared band in each image was ignored). The sample images used to populate our database were acquired from four different geographic locations, each having its own characteristic landscape (see Table 1). The Moscow area, for example, contains many diverse landcover types in every 512 × 512 subimage that was extracted. These landcover types include coniferous forest, deciduous forest, and agriculture. The Moscow images look nothing like the images around the other three geographic locations. Similarly, the Cairo landscape is unique and dissimilar to the Moscow, Albuquerque, and Los Alamos areas.

We computed global spectral signatures for each database image by clustering the 6-dimensional pixel vectors into 20 clusters. By assuming that each cluster represents a set of data that could be generated by a Gaussian random process, we can compute the maximum likelihood parameters for that Gaussian random process.[11] The estimates for the mean vector and covariance matrix for the Gaussian associated with the i^{th} cluster (containing

LOCATION	DOMINANT LANDSCAPE COVER
Moscow (Russia)	Coniferous Forest, Deciduous Forest, Agriculture, ...
Cairo (Egypt)	Agriculture, Dense Urban, ...
Albuquerque (USA)	Desert, Coniferous Forest, ...
Los Alamos (USA)	Desert, Coniferous Forest, ...

Table 1: Selected Geographic Locations

M_i pixels) are found using the equations below:

$$\underline{\mu}_i = \frac{1}{M_i}\sum_{k=1}^{M_i}\underline{x}_k \qquad \Sigma_i = \frac{1}{M_i}\sum_{k=1}^{M_i}\left(\underline{x}_k - \underline{\mu}_i\right)\left(\underline{x}_k - \underline{\mu}_i\right)^T \tag{18}$$

The resultant Gaussian distribution functions are weighted by the percentage of image pixels that were assigned to the associated clusters, and the overall PDF is taken to be the weighted sum of these 20 Gaussians. It can be shown that the maximum likelihood estimates for a Gaussian that is subject to the axis alignment constraint described in Section 3 can be found by simply setting the off-diagonal elements of Σ_i to 0. Similarly, when enforcing the hyperspherical constraint described in Section 4, we can simply compute the average of the N diagonal elements of Σ_i to get σ_i^2.

Initial timing results indicate that we get about a 5× speedup when using axis alignment constraints, and an 11× speedup when using the hyperspherical constraints for performing searches on this data set. Computing 120 inner products usually takes 6.15 CPU seconds, but it takes only 1.22 CPU seconds when using the axis constraints, and 0.56 CPU seconds when using the hyperspherical constraints. We have observed that, as expected, the improvement in speed increases with the dimensionality of the feature space. Timings were obtained on a Sun SPARCStation 20.

If we look at how the different constraints affect the order that database images are retrieved, we see that some things do indeed change. It is difficult to judge, however, whether this constitutes a degraded performance in the ability of **CANDID** to compare images. Figure 2 shows the sorted retrieval scores (normalized inner-product values) when using one of the Moscow images as a query image. All three signature representation techniques produce data sets where all Moscow images are retrieved from the database before any of the other (Albuquerque, Los Alamos, Cairo) images. The specific order that these Moscow images are retrieved, however, does change. Since image similarity is a somewhat subjective measure, we might say that as long as all Moscow images are retrieved first, then the results obtained by our three methods are comparable.

6 Conclusions

We have successfully increased the speed of comparing **CANDID** signatures by an order of magnitude. This was done by enforcing constraints on the shape and orientation of individual Gaussian components. Since the "similarity" between two images is a subjective measure, the results obtained when using our constraints may or may not affect the retrieval process adversely. If it turns out that these constraints affect results too much for general search and retrieval purposes, they may still provide a good starting point for making a "first pass" during a database query. In this scenario, we would use the faster signature comparison methods to discount database images that would not be expected to produce significant similarity scores. The original, more expensive, comparison techniques would then be used to sort the remaining images.

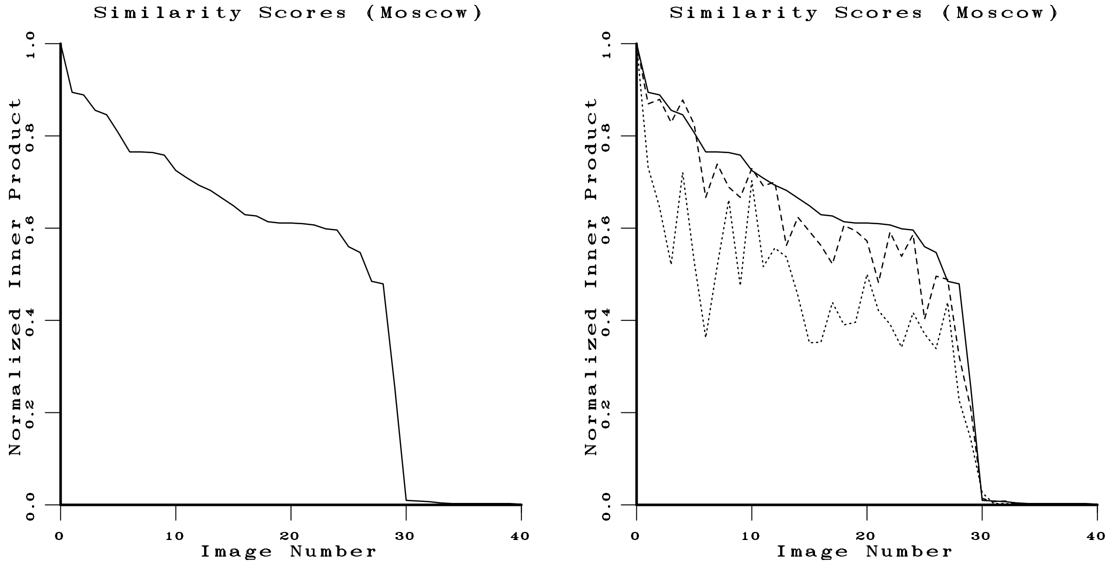

Figure 2: Moscow Retrieval Scores (Spectral Features / 20 Gaussians). The plot on the left shows the sorted similarity scores between the Moscow query image and the top 40 matches in the database. All images other than the first 30 produced negligible similarity scores; only the first few are depicted in the plot. The plot on the right also shows the similarity scores for those 40 database images when using axis constraints (— — —), as well as hyperspherical constraints (···).

7 Acknowledgements

We gratefully acknowledge Clint Scovel's contributions to technical discussions related to probability density function representation and manipulation. This work was performed under a U.S. Government contract (W-7405-ENG-36) by Los Alamos National Laboratory, which is operated by the University of California for the U.S. Department of Energy.

8 REFERENCES

[1] C. Faloutsos, M. Flickner, W. Niblack, D. Petkovic, W. Equitz, and R. Barber. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, pages 231–262, 1994.

[2] J. Ashley, R. Barber, M. Flickner, J. Hafner, D. Lee, W. Niblack, and D. Petkovic. Automatic and semi-automatic methods for image annotation and retrieval in qbic. In *SPIE Vol. 2420 Storage and Retrieval for Image and Video Databases III*, pages 24–35, 1995.

[3] M.J. Swain and D.H. Ballard. Color indexing. *International Journal of Computer Vision*, 7(1):11–32, 1991.

[4] H.S. Sawhney and J.L. Hafner. Efficient color histogram indexing. In *Proceedings of the 1994 IEEE International Conference on Image Processing*, volume 2, pages 66–70, 1994.

[5] M. Stricker and M. Orengo. Similarity of color images. In *SPIE Vol. 2420 Storage and Retrieval for Image and Video Databases III*, pages 381–392, 1995.

[6] P.M. Kelly, T.M. Cannon, and D.R. Hush. Query by image example: The CANDID approach. In *SPIE Vol. 2420 Storage and Retrieval for Image and Video Databases III*, pages 238–248, 1995.

[7] J. Barros, J. French, W. Martin, P. Kelly, and M. Cannon. Using the triangle inequality to reduce the number of comparisons required for similarity-based retrieval. In *SPIE Vol. 2670 Storage and Retrieval for Still Image and Video Databases IV*, 1996.

[8] R.E. Kimbrell. Searching for text? send an n-gram! *BYTE*, pages 297–312, May 1988.

[9] T.R. Thomas. Document retrieval from a large dataset of free-text descriptions of physician-patient encounters via n-gram analysis. Technical Report LA-UR-93-0020, Los Alamos National Laboratory, Los Alamos, NM, 1993.

[10] M. Damashek. Gauging similarity via n-grams: Language-independent sorting, categorization, and retrieval of text. *Science*, 267(5199), February 1995.

[11] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley, New York, NY, 1973.

[12] J.T. Tou and R.C. Gonzalez. *Pattern Recognition Principles.* Addison-Wesley, Reading, MA, 1974.

[13] P.M. Kelly, D.R. Hush, and J.M. White. An adaptive algorithm for modifying hyperellipsoidal decision surfaces. *Journal of Artificial Neural Networks*, 1(4):459–480, 1994.

[14] P.M. Kelly. An algorithm for merging hyperellipsoidal clusters. Technical Report LA-UR-94-3306, Los Alamos National Laboratory, Los Alamos, NM, 1994.